Définition et manipulation des données

Définition et manipulation des données

1. Création de tables

2. Insertion des données

3. Modification des données

3. Suppression des données

4. Définition des contraintes

Création de table

```
CREATE TABLE table (liste de définition de colonnes);

table::= nom simple d'une table

définition de colonne::= nom de colonne(type)
[liste de contraintes de colonne]
```

Création de table

Exemple: Création de la table COMPAGNIE

comp	nrue	rue	ville	nomComp

INSTRUCTION SQL

```
CREATE TABLE compagnie
(comp VARCHAR2(4),
 nrue NUMBER(3),
 rue VARCHAR2(20),
 ville VARCHAR2(15),
 nomComp VARCHAR2(15));
```

Création de table

Exemple: Création de la table COMPAGNIE avec 2 contraintes:

- Une valeur par défaut `'PARIS'` pour ville
- Une valeur non nulle pour nomComp

INSTRUCTION SQL

```
CREATE TABLE compagnie
(comp VARCHAR2(4),
 nrue NUMBER(3),
 rue VARCHAR2(20),
 ville VARCHAR2(15) DEFAULT 'PARIS',
 nomComp VARCHAR2(15) NOT NULL);
```

Commentaires

La table contient 5 colonnes (4 chaînes de caractères et une valeur numérique de 3 chiffres)
2 contraintes:

- **DEFAULT** qui fixe PARIS comme valeur par défaut de la colonne *ville*
- **NOT NULL** qui impose une valeur non nulle dans la colonne *nomComp*.

Création de table: contraintes

Le SGBD vérifie les contraintes lors des insertions ou mises à jour de n-uplets

- Quand créer les contraintes?
  - Lors de la création de la table
  - Lors d'une évolution ultérieure du schéma de la table
- Les contraintes définissent **des règles de gestion** au niveau des colonnes des tables.
- Quels types de contraintes?
  - Unicité
  - Autorisation des valeurs nulles
  - Clé primaire (PK, pour Primary Key)
  - Clé étrangère (FK, pour Foreign Key)
  - Valeurs autorisées (check)

Création de table: contrainte de colonne

- Clause de définition d'une colonne

```
colonne type [DEFAULT exp]
             [ [CONSTRAINT nom] def_contrainte_colonne]
```

- Contrainte de colonne (nommées et non nommées)

```
NULL | NOT NULL
    ou
UNIQUE | PRIMARY KEY
ou
REFERENCES table [(nom_de_colonne)]
ou
CHECK (condition_sur_valeur)
```

Création de table: contraintes de Colonne

- **Explication des contraintes**

Contrainte	Définition
NULL/NOT NULL	Autorise (NULL) ou interdit (NOT NULL) l'insertion de valeur NULL pour cet attribut
PRIMARY KEY	Désigne l'attribut comme clé primaire de la table. Cette contrainte ne peut apparaître qu'une seule fois dans l'instruction
`REFERENCES table [(nom_de_colonne)]`	Contrainte d'intégrité référentielle pour l'attribut dans la table. Les valeurs prises par cet attribut doivent exister dans l'attribut nom_de_colonne .
CHECK (condition)	Vérifie lors de l'insertion de lignes que l'attribut réalise la condition
DEFAULT	Permet de spécifier une valeur par défaut pour toutes les lignes, lorsque la vraie valeur n'est pas donnée.
UNIQUE	Deux tuples de la table ne peuvent pas avoir la même valeur pour cette colonne. Implicite s'il y ' a une contrainte de clé primaire sur la colonne.

Déclaration des contraintes

Deux types de contraintes:

1. Contraintes en ligne: déclarées au même temps que la colonne

2. Contraintes nommées: utilisent le mot-clé CONSTRAINT suivi d'un identifiant et de la définition de la contrainte

REMARQUE: ORACLE recommande de déclarer les contraintes NOT NULL en ligne, les autres peuvent être déclarées soit en ligne, soit nommées.

Contrainte de clé primaire

- **Contrainte nommée**

```
CONSTRAINT nomc PRIMARY KEY(atti,attj, ...)
```

- Une contrainte de clé primaire indique que l'ensembles des attributs (atti,…, attj) sert d'identifiant pour les n-uplets de la table

- Implique NOT NULL et UNIQUE sur chacun des (atti,…,attj)

- Au Maximum une contrainte PRIMARY KEY par table

Contrainte de clé primaire

Exemple:

Création d'une table tablePK avec 2 attributs (att1 et att2) tel que att1 de type NUMBER est une clé primaire

```
CREATE TABLE tablePK
(att1 NUMBER, att2 NUMBER, CONSTRAINT pk_tablePK PRIMARY KEY(att1));
```

OU

```
CREATE TABLE tablePK
(att1 NUMBER PRIMARY key,
 att2 NUMBER);
```

Contrainte de clé primaire

Exemple: Création de **la table COMPAGNIE** avec comp comme clé primaire

comp	nrue	rue	ville	nomComp

INSTRUCTION SQL (avec contrainte nommée)

```
CREATE TABLE Compagnie
(comp VARCHAR2(4) CONSTRAINT PK_Compagnie
PRIMARY KEY,
 nrue NUMBER(3),
 rue VARCHAR2(20),
 ville VARCHAR2(15) DEFAULT 'PARIS',
 nomComp VARCHAR2(15) NOT NULL);
```

INSTRUCTION SQL (sans noms de contraintes)

```
CREATE TABLE Compagnie
(comp VARCHAR2(4) PRIMARY KEY,
 nrue NUMBER(3),
 rue VARCHAR2(20),
 ville VARCHAR2(15) DEFAULT 'PARIS',
 nomComp VARCHAR2(15) NOT NULL);
```

Contrainte de clé étrangère

```
CONSTRAINT nomc FOREIGN KEY(atti,…, attj)
REFERENCES table_cible (att'1,…, att'j)
```

- Une clé étrangère est une référence vers la clé primaire d'une autre table
- Les valeurs pour (att1,..,attj) doivent correspondre aux valeurs d'un des n-uplets de table_cible pour ses attributs (att'1,..,att'j)

Contrainte de clé étrangère

Exemple:

Création d'une table tableFK avec deux attributs (att et att1) de type NUMBER, le premier est clé primaire et le second est étrangère faisant référence à att1 de la table tablePK.

```
CREATE TABLE tableFK
(att NUMBER PRIMARY KEY,
 att1 NUMBER,
CONSTRAINT Fk_tableFK_att1 FOREIGN KEY(att1) REFERENCES tablePK(att1));
```

OU

```
CREATE TABLE tableFK
(att NUMBER PRIMARY KEY,
att1 NUMBER CONSTRAINT Fk_tableFK_att1 REFERENCES tablePK);
```

OU

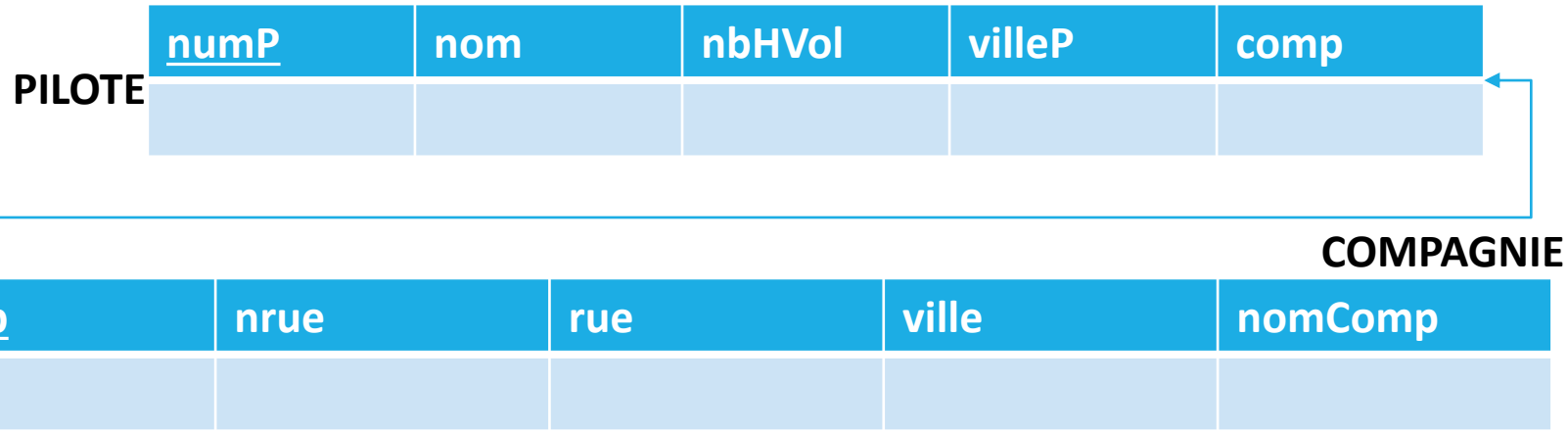
```
CREATE TABLE tableFK
(att NUMBER PRIMARY KEY,
att1 NUMBER REFERENCES tablePK(att1));
```

Contrainte de clé étrangère: Exemple

Créer la table pilote avec:

-numP: clé primaire

-Comp: clé étrangère



INSTRUCTION SQL

```
CREATE TABLE Pilote
(numP NUMBER(4) CONSTRAINT Pk_Pilote PRIMARY KEY,
 nom VARCHAR2 (15),
 nbHVol NUMBER(7,2),
 villeP VARCHAR2 (10),
 comp VARCHAR2(4) CONSTRAINT FK_Pilote_comp REFERENCES Compagnie);
```

Contrainte check

```
CONSTRAINT nomc CHECK (condition)
```

La contrainte check permet de spécifier une condition (booléenne) qui sera vérifiée lors d'insertions ou modification de n-uplets

Exemple:

```
CREATE TABLE tableCheck
(att1 NUMBER,
 att2 NUMBER,
CONSTRAINT verif_att CHECK(att1>1));
```

OU

```
CREATE TABLE tableCheck
(att1 NUMBER CHECK(att1>1),
 att2 NUMBER);
```

Création d'une table tableCheck avec deux attributs. Le premier a une contrainte qui force ses valeurs à être supérieures à 1.

Création de table: Exemple

Exemple:

Créer la table Avion avec:

- NumAv: clé primaire

- capacité d'avion (CapAv) supérieure ou égale à 4

- comp : clé étrangère vers Compagnie

Avion

numAv	AvNom	nbHVolA	capAv	Loc	comp

INSTRUCTION SQL

```
CREATE TABLE Avion
(NumAv NUMBER(4)  CONSTRAINT PK_Avions PRIMARY KEY,
 AvNom VARCHAR2(20),
 NBHVOLA NUMBER(4),
 CapAV NUMBER(4) CONSTRAINT C_CapAvions CHECK(CapAv >= 4),
 Loc VARCHAR2(15),
 comp VARCHAR2(4) CONSTRAINT FK_Avion_comp REFERENCES Compagnie );
```

Création de table: Exemple

Vol

numVol	numP	numAv	dateA	dateD	villeA	villeD

Exemple

Créer la table vol en prenant considération les deux contraintes suivantes:

C1) Pour un vol, sa ville de départ est différente de sa ville d'arrivée.

C2) Pour un vol, son heure de départ est strictement antérieure à son heure d'arrivée.

INSTRUCTION SQL

```
CREATE TABLE vol
       (numVol NUMBER(4)  CONSTRAINT PK_Vol PRIMARY KEY,
        numP NUMBER(4) CONSTRAINT FK_Pilote_num REFERENCES Pilote,
        NumAv NUMBER(4) CONSTRAINT FK_Avion_num REFERENCES Avion,
         VilleD CHAR(10) NOT NULL,
         VilleA CHAR(10) NOT NULL,
         DateD DATE,
         DateA DATE,
         CONSTRAINT C1_Vols CHECK(VilleD <> VilleA),
         CONSTRAINT C2_Vols CHECK(DateD < DateA));
```

Création de table: résumé

- L'ordre de création des tables est important quand on définit des contraintes en même temps que les tables.

- Il faut d'abord créer les tables « pères » puis les tables « fils ».

- PRIMARY KEY = UNIQUE + NOT NULL+ index

Insertion des lignes dans une table

```
INSERT INTO table [ (col,...,col) ]{ VALUES (val,..,val) | requête }
```

1. Ajouter une compagnie dans la table COMPAGNIE en respectant l'ordre de définition des colonnes :

```
INSERT INTO Compagnie
  VALUES ('SING',  7, 'Camparols', 'Singapour','Singapore AL');
```

COMP	NRUE	RUE	VILLE	NOMCOMP
SING	7	Camparols	Singapour	Singapore AL

Insertion des lignes dans une table

2. Ajouter une compagnie dans la table COMPAGNIE sans connaître l'ordre de définition des colonnes

```
INSERT INTO Compagnie (comp, nrue,rue, nomComp )
  VALUES ('AC',8, 'Champs Elysées','Castanet Air'); //DEFAULT implicite
```

COMP	NRUE	RUE	VILLE	NOMCOMP
SING	7	Camparols	Singapour	Singapore AL
AC	8	Champs Elysées	PARIS	Castanet Air

3. Ajouter une Compagnie dont le numéro de rue est inconnu

```
INSERT INTO Compagnie (comp, rue, ville,nomComp )
  VALUES ('AN2', 'Foch', 'Balgnac', 'Air Nul2');
```

COMP	NRUE	RUE	VILLE	NOMCOMP
SING	7	Camparols	Singapour	Singapore AL
AC	8	Champs Elysées	PARIS	Castanet Air
AN2	(null)	Foch	Balgnac	Air Nul2

Insertion des lignes dans une table: exemple

Résultat de l'insertion

comp	nrue	rue	ville	nomComp
SING	7	Camparols	Singapour	Singapore AL
AC	8	Champs Elysées	Paris	Castanet Air
AN2	NULL	Foch	Balgnac	Air Nul2

Valeur par défaut

Modification de table/colonne

RENOMMER UNE TABLE: RENAME TO

Commande: ALTER TABLE

- Ajout de colonne (ADD)
- Renommer une colonne (RENAME COLUMN)
- Modification de colonne (MODIFY)
- Suppression d'une colonne (DROP COLUMN)
- Ajout/suppression/ modification des contraintes (ADD CONSTRAINT/ DROP CONSTRAINT)

Modification de table

- **Renommer** une table

```
RENAME table-Name TO new-Table-Name;
```

- **Ajouter** une ou plusieurs colonnes de table

```
ALTER TABLE nom_table
ADD ( {def_colonne}  [, {def_colonne} ] ... );
```

Modification de colonne

PILOTE

numP	nom	nbHVol	villeP	comp

NUMP	NOM	NBHVOL	VILLEP	COMP	AGE
200	PATRICK	20	Marseille	AF	(null)
300	TEST	30	Paris	SING	(null)
500	MARC	15	Paris	AF	(null)
140	jacques	(null)	Marseille	AN2	(null)
160	MARC	20	Nice	SING	(null)

- **Ajout de colonnes à la table Pilote**

```
ALTER TABLE Pilote ADD (Age NUMBER(2));
```

Insère une colonne Age en l'initialisant à NULL pour tous les pilotes

- **Renommer la colonne ville en adresse**

```
ALTER TABLE Pilote RENAME COLUMN VilleP TO adresse;
```

NUMP	NOM	NBHVOL	ADRESSE	COMP	AGE
200	PATRICK	20	Marseille	AF	(null)
300	TEST	30	Paris	SING	(null)
500	MARC	15	Paris	AF	(null)
140	jacques	(null)	Marseille	AN2	(null)
160	MARC	20	Nice	SING	(null)

Modification de colonne et de contrainte

```
ALTER TABLE nom_table
MODIFY ( def_colonne [, def_colonne ] ... );
```

PILOTE

numP	nom	nbHVol	villeP	comp

Exemple:

- **Modifier le type de la colonne:**
- Modifier le type de la colonne nom de VARCHAR2 en CHAR tout en déclarant NOT NULL.

```
ALTER TABLE Pilote MODIFY nom CHAR(4) NOT NULL;
```

Attention: -Si vous changez le type de la colonne vers un type différent (e.g de NUMBER→ CHAR), , il faut vider la colonne
- MODIFY permet de modifier le type des données et de définir des contraintes

Modification de contrainte

```
ALTER TABLE nom_table
ADD CONSTRAINT nomc   (def_contrainte);
```

Exemple:

Ajout d'une contrainte sur l'âge des pilotes qui ne doit pas dépasser 45 ans

```
ALTER TABLE Pilote ADD CONSTRAINT limit_age CHECK(age<45);
```

Modification de table

- **Supprimer une ou plusieurs contraintes de table**

```
ALTER TABLE table
DROP CONSTRAINT nomc
[DROP CONSTRAINT nomc] ...;
```

- **Supprimer des colonnes**

```
ALTER TABLE table
DROP COLUMN nom_colonne;
```

Attention: -Vous ne pouvez pas supprimer une colonne primaire si elle est référencée dans d'autres tables
- Une colonne peut être supprimée même si elle contient des données

Suppression de contrainte

```
ALTER TABLE nom_table DROP PRIMARY KEY;
```

Cette commande permet de supprimer la contrainte de clé primaire de la table

OU

```
ALTER TABLE nom_table DROP CONSTRAINT nom_c;
```

Cette commande permet de supprimer une contrainte dont le nom est nom_c

Consultation de la structure de table

```
DESCRIBE nom_table
```

Cette commande indique le nom et le type de données de chaque colonne de la table et indique aussi si l'absence de valeur est autorisée ou interdite.

Exemple:

```
DESCRIBE Compagnie
```

Résultat sous Oracle SQL Developer:

```
Nom      NULL ?   Type
-------  -------- ------------
COMP     NOT NULL VARCHAR2(4)
NRUE              NUMBER(3)
RUE               VARCHAR2(20)
VILLE             VARCHAR2(15)
NOMCOMP  NOT NULL VARCHAR2(15)
```

Modification des données

La commande UPDATE permet de modifier les valeurs d'une ou plusieurs colonnes, dans une ou plusieurs lignes existantes d'une table. La syntaxe est la suivante :

```
UPDATE nom_table
SET nom_col_1 = {expression_1 | ( SELECT ...) },
    nom_col_2 = {expression_2 | ( SELECT ...) },
    ...
    nom_col_n = {expression_n | ( SELECT ...) }
WHERE predicat
```

Les valeurs des colonnes nom_col_1, nom_col_2,… nom_col_n sont modifiées dans toutes les lignes qui satisfont le prédicat predicat. En l'absence d'une clause WHERE, toutes les lignes sont mises à jour. Les expressions expression_1, expression_2,… expression_n peuvent faire référence aux anciennes valeurs de la ligne.

Modification des données

1. Modification de la compagnie de code ' SING' en affectant la valeur 50 à la colonne nrue.

```
UPDATE Compagnie SET nrue=50 WHERE comp='SING'
```

COMP	NRUE	RUE	VILLE	NOMCOMP
SING	50	Camparols	Singapour	Singapore AL

2. Modification de plusieurs colonnes: modification de la compagnie de code 'AN2' en affectant simultanément la valeur 14 à la colonne nrue et la valeur par défaut (‘Paris‘) à la colonne Ville

```
UPDATE Compagnie SET nrue=14, ville=DEFAULT WHERE comp='AN2'
```

Modification des données

Table après les modifications:

comp	nrue	rue	ville	nomComp
SING	50	Camparols	Singapour	Singapore AL
AC	8	Champs Elysées	Paris	Castanet Air
AN2	14	Foch	Paris	Air Nul2

Destruction de table

```
DROP TABLE table [CASCADE CONSTRAINTS]
```

Quand une table est supprimée, Oracle :

- efface tous les index qui y sont attachés quelque soit le propriétaire
- efface tous les privilèges qui y sont attachés

Vérification de la cohérence par le SGBD:

Que faire des lignes qui font références à celles supprimées ?

- Le programmeur peut le spécifier via l'option CASCADE CONSTRAINTS : les contraintes liant des lignes à celles supprimées, sont aussi supprimées.
- Si le programmeur ne spécifie rien, alors la suppression n'est pas effectuée dès lors qu'elle affecte d'autres objets.

Destruction de table

Exemple: Supprimer la table compagnie

```
DROP TABLE compagnie
```

```
Erreur commençant à la ligne: 1 de la commande -
drop table compagnie
Rapport d'erreur -
ORA-02449: unique/primary keys in table referenced by foreign keys
02449. 00000 -  "unique/primary keys in table referenced by foreign keys"
*Cause:    An attempt was made to drop a table with unique or
           primary keys referenced by foreign keys in another table.
*Action:   Before performing the above operations the table, drop the
           foreign key constraints in other tables. You can see what
```

```
DROP TABLE compagnie CASCADE CONSTRAINTS
```

```
Table COMPAGNIE supprimé(e).
```

PILOTE

numP	nom	nbHVol	villeP	comp

COMPAGNIE

comp	nrue	rue	ville	nomComp

Suppression des lignes

```
DELETE [FROM] table [alias]
[WHERE condition]
```

1. Vider toute la table COMPAGNIE

```
DELETE FROM Compagnie;
```

2. Supprimer tous les pilotes de la compagnie de code ‘SING’

```
DELETE FROM Pilote WHERE comp='SING' ;
```

En résumé

Des commandes de définition des données:

- Création de tables: CREATE TABLE
- Insertion des données: INSERT INTO
- Modification de table: ALTER TABLE
- Mise à jour : UPDATE
- Suppression de lignes: DELETE FROM
- Suppression de table: DROP

Synthèse

SQL: un langage de définition des données

- CREATE, DROP, ALTER

SQL: un langage de manipulation des données

- SELECT, INSERT, DELETE, UPDATE

Références

Livres:

G. Gardarin : Bases de données objet et relationnel. Eyrolles ed. 1999.

R. Elmasri et S.Navathe: Conception et architecture des bases de données. Pearson Education

C.Soutou. SQL pour oracle Applications avec Java, PHP et XML, edition Eyrolles

Supports de cours:

- Support de cours : C. Sabatier, Aix Marseille Université
- Support de cours : C.Capponi, Aix Marseille Université
- Support de cours: O.Papini, Aix Marseille Université (http://odile.papini.perso.luminy.univ-amu.fr/sources/BD.html)

Sites Web

Site officiel d'Oracle: https://docs.oracle.com/database/121/

Fonctions SQL Oracle: https://docs.oracle.com/cd/B19306_01/server.102/b14200/functions001.htm